Connecting to the MC³ from EtherNet/IP via ABC-EIP

Using EtherNet/IP as a transport between MC³ controllers and PLC's offers cost savings, improved diagnostic capabilities, better accuracy and tighter control. A single EtherNet/IP network replaces numerous analog and digital I/O modules, eliminates analog signal calibration and ensures data integrity. Once the network is in place, adding I/O points and variables for monitoring is a matter of configuration rather than pulling cables and buying new I/O modules.

The configuration described in this document is fully supported by Merrick Industries. There is equipment on hand here to test user specific configurations and to deliver pre-configured and tested solutions.

INTRODUCTION

It is possible to exchange status and variables between MC³ controllers and EtherNet/IP enabled PLCs, using an ABC-EIP protocol converter from HMS Industrial Networks AB, (EIP). Communications replaces both analog and digital I/O. Full Floating Point accuracy is maintained for variables. Up to 10 variables and 80 I/O points can be configured per MC³ controller.

Newer versions of Merrick's MC³ controller support ModBus RTU serial communications and exposes a standardized Common Interface Table (CIT). ModBus functions 3 (Read Holding Registers), 16 (Preset Multiple Regs) and 8, sub function 0 (Return Query Data) are supported.

The EIP has an EtherNet/IP interface one side and a Modbus Master interface on the other. Many MC³ controllers can be connected to the EIP using the existing 4-wire RS-422 interface on the EIP and the existing 4-wire RS-485 interface on the MC³ controller.

The information in this document applies to the following MC³ firmware versions:

Firmware	Used for	Released	Comm Ver
20.10.EX.F	Belt Feeder	03/28/02	1
20.20.EX (All)	Belt Feeder	04/17/03	2
24.10.EX.H and later	Pressurized Coal Feeder	08/02/02	1
30.00.EX.C and later	Loss-In-Weight	04/25/02	1
30.10.EX.E and later	Enhanced Loss-In-Weight	06/06/04	2
35.00.EX.B and later	Weigh-Out Batcher		2
40.10.EX.A and later	Impact Flow Meter	04/14/03	2
90.10.EX.Y and later	MasterSet	01/02/03	2

Other Merrick firmware releases may also support ModBus RTU communications.

The EIP maintains a conversation with many MC³ controllers, using Modbus RTU, and then exposes one aggregated read and one aggregated write table to the PLC. For real time control purposes, a maximum of 4 MC³s per EIP can be used. For monitoring only, up to 16 MC³'s can be connected.

Data tables are transferred between the MC³s and the EtherNet/IP host PLC. The positioning and content of the data elements in the tables must be tracked all the way from the internal MC³ register database to the data structures in the PLC. This is done in several steps:

- Between the MC³ register database and the MC³ CIT. Some of this mapping is fixed, and some is configurable. You set this up in the MC³, in the Comm 1 Numeric screen.
- Between the MC³ CIT and the Read and Write Tables in the EIP. This is entirely configured in the EIP, using the ABC Configuration Utility.
- Between the EIP tables exposed to EtherNet/IP and the adapter data structure in the PLC. This is done in the PLC I/O configuration, or by using Message instructions.

- Between the adapter data structure in the PLC and the final data structures in the PLC. This is done with PLC programming methods.

For performance reasons, all network transfers encompasses entire tables. For this reason, all elements in a table transfer must be of the same data type. Modbus RTU transfers handles 16 bit INT's, by design. EnterNet/IP transfers handles 32 bit DINT's in this example.

In the CIT, and eventually in the PLC, different data types are used for control/status bits, integer numbers and floating-point numbers. Control/Status bits and integer numbers are organized in INTs (16 bit words). Parameters are organized as REALs (IEEE 32 bit floating point numbers), located in two consecutive 16 bit words. In order to have usable data in the PLC, a user defined data structure is used, which closely resembles the MC³ CIT table.

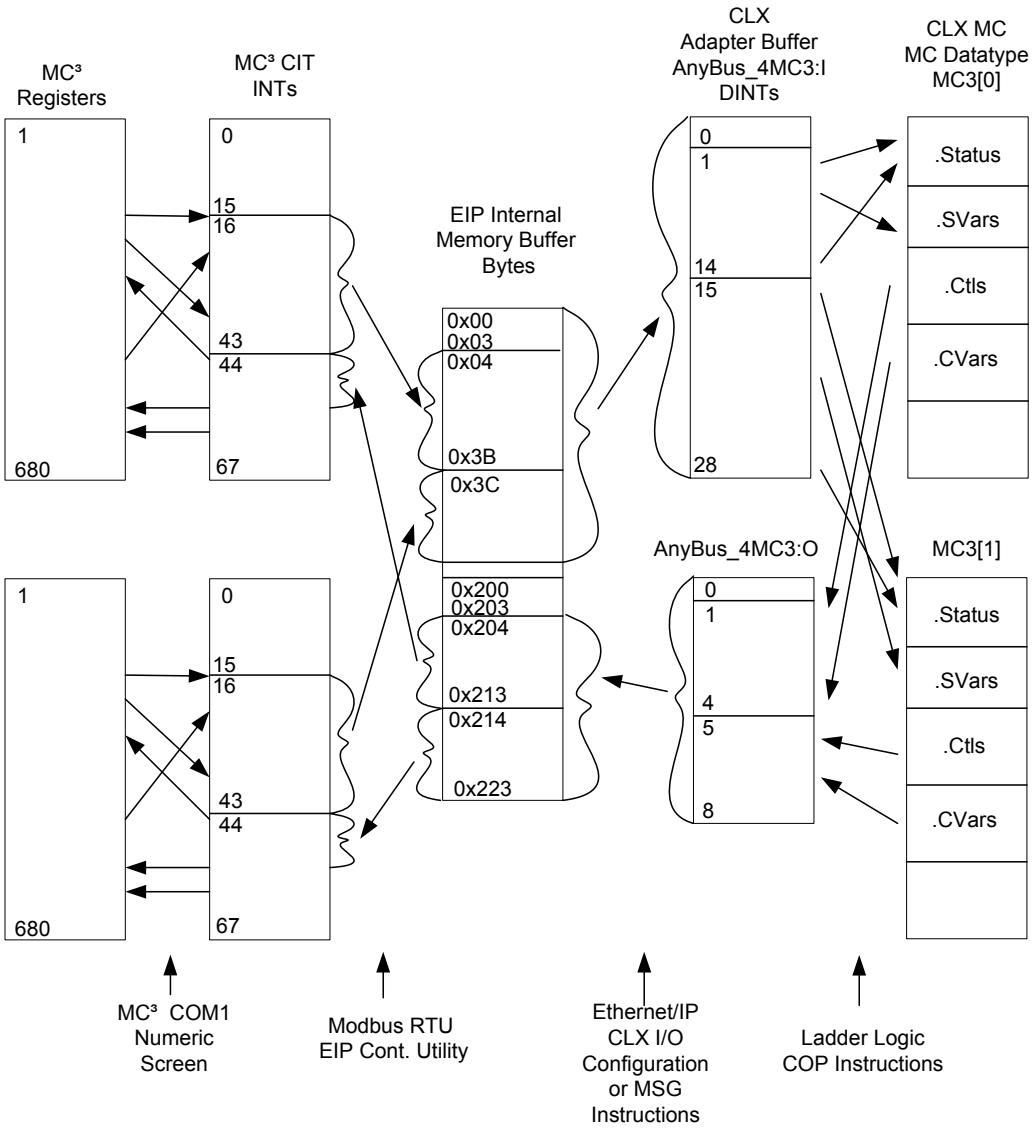


Table transfer between MC³ and the PLC MC[n] data structures. This block diagram shows a two MC³ set-up.

CONFIGURATION EXAMPLE

We used one Allen-Bradley ControlLogix PLC (CLX), one EIP and four MC³ controllers. A "Read All, Write some" approach was taken, meaning that as much information as possible was transferred from the MC³s to the PLC, and just enough was going the other way.

The CLX was configured to accept the EIP as an I/O device on the EtherNet/IP Network. Some logic was added to monitor the EIP status, to safeguard communications integrity and to extract and insert the MC³ data out of and in to useful data structures.

The EIP interface was configured to maintain a dialogue with four MC³ controllers, aggregate the data and expose a Read and Write area to EtherNet/IP.

The MC³ controllers were configured to accept control data and expose data for monitoring using Modbus RTU.

In the end, the MC³ controllers become an integral part of the CLX I/O, with an easy to use programming interface, in the form a single tag.

Equipment used:

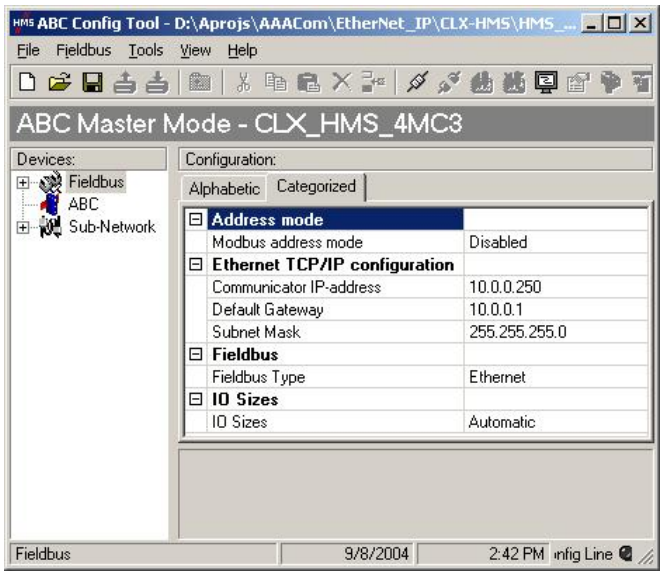
- Laptop PC, DELL Inspiron 8600.
  MS Windows XP Professional, SP1.
  Rockwell Software RSLogix 5000. V10.00.00.
  Rockwell Software RSLinx V2.41.00.
  HMS Networks ABC Configurator version 2.00 rev 4.
- Allen-Bradley 1756-L55 ControlLogix 5555 PLC with 1756-ENBT/A EtherNet adapter (CLX).
- NetGear EtherNet hub.
- ABC-EIP EtherNet/IP Interface, order code AB7007 (EIP).
- Patch Cables.
- Cable for configuring the EIP from the PC, HMS part number 017620.
- 24 V DC Power Supply.
- Home made serial interface cable ABC-EIP <-> MC³ controllers. See "Connect the EIP to the MC³s and check communications", on page 10.

Configuring the ABC-EIP

EIP configuring is described in the AnyBus Communicator User Manual (ABCUM). The ABC Configurator and ABCUM can be downloaded from the HMS web site, http://www.anybus.com. We are trying to keep specific links updated on the Merrick connectivity web site http://merrick-inc.com/mct. You can also download the actual ABC configuration file we used from that site.

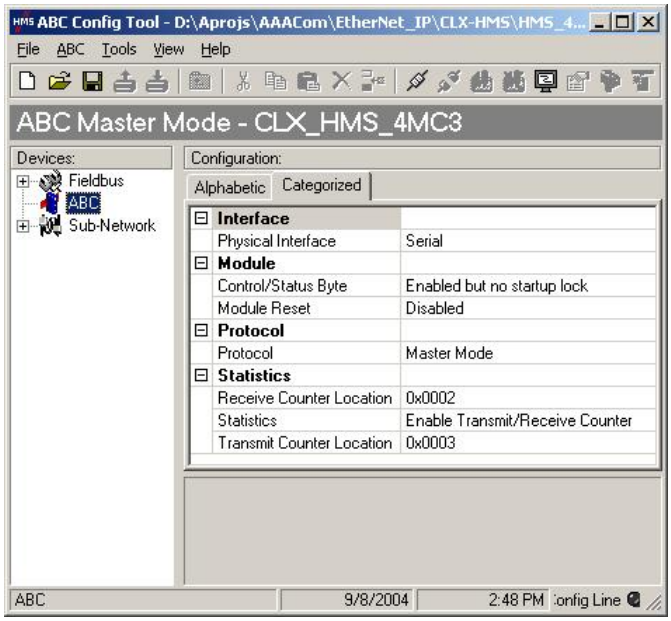
The EIP requires 24 V DC. Connect according to ABCUM, appendix C. A serial interface cable must be purchased or manufactured. Details can be found in ABCUM, appendix C, “PC Connector”. Connect 24V and the PC Connector cable, and start the configurator. Configuration is described in ABCUM, starting with chapter 4. You can skip Generic Data Mode (chapter 5). You will be using Master Mode, described in chapter 6.

Fieldbus



Enter your IP address and subnet mask. Make sure the CLX and the EIP are on the same subnet. If you don’t intend to route the traffic, use 0.0.0.0 as the Default Gateway IP address.

ABC



Control/Status byte should be set to “Enable but no startup lock”. This enables you to check the RTU communication (with the MC³’s) without having EtherNet/IP connected.

The "Module Reset" should be set to "Disabled". The function of this setting is obscure.

The Protocol setting must be “Master Mode”, meaning Modbus RTU master.

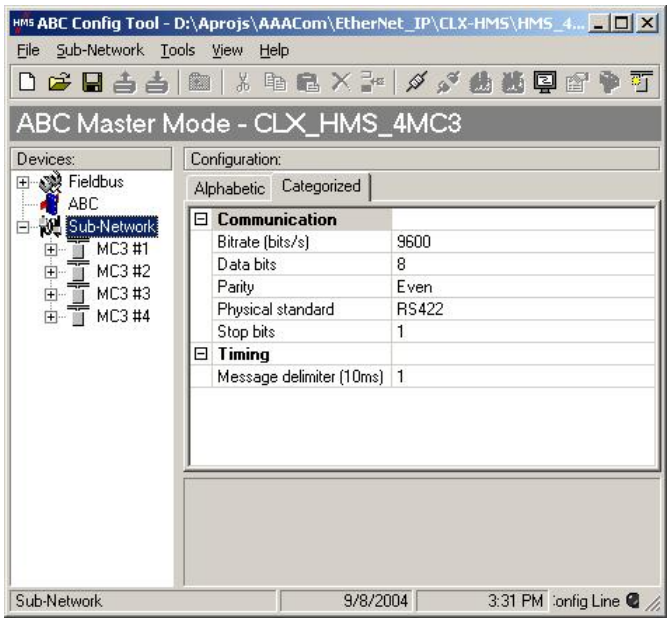
Statistics are useful for Modus RTU troubleshooting. Bytes 2 and 3 are unused and can eventually be inspected from the PLC

Sub-Network

The "Sub-Network" is in reality the multi-drop Modbus RTU network used to connect the MC³ controllers to the EIP. This is where almost all configuration takes place. There are settings for the overall Sub-network, for the individual slaves (MC³'s in this case), for the actions for each slave and for telegram parameters for each action.

The Sub-network settings are global and should be set as follows for successful MC3 operation:

Parameter	Settings	Comment
Bitrate	9600 or 19200	ModBus RTU standard is 9600.
Data Bits	8	Mandatory Modbus RTU setting.
Parity	Any	'Even' is Modbus RTU standard.
Physical Standard	RS422	This is a four wire RS485 hook up. From the Modbus Master's point of view, RS422 is the correct setting.
Start Bits	1	There are actually no options.
Stop Bits	1 or 2	Use 1.
Message Delimiter	1 (10 ms)	Set to at least 1. There is no performance gain in setting this parameter to 0.



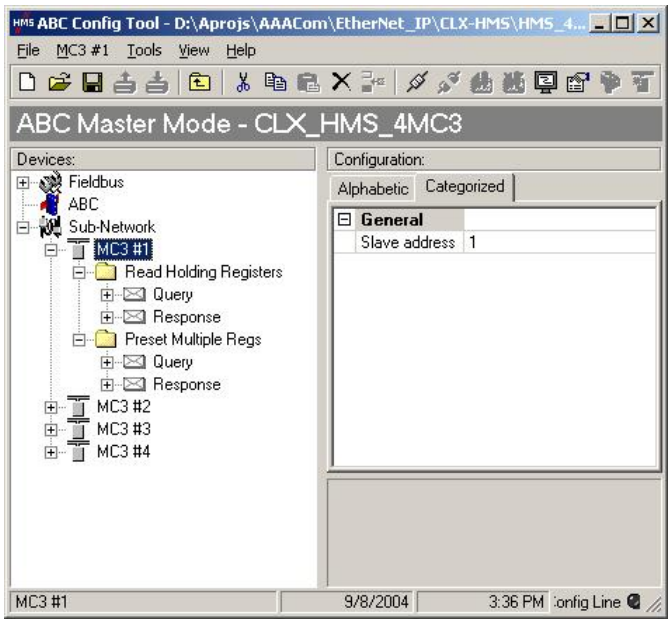
All MC³ serial parameter settings must agree with the parameters in this table. Newer MC³ controllers support four wire RS485 and RS232, without re-configuration. For older controllers, and if you connect to more than one controller, you must use RS422.

Under the Sub-Network level, insert nodes, one per MC³ you want to connect. This is a screen dump of our 4 node setup.

The slave Address has to be different for each node and must be equal to the 'Controller Number' setting in the corresponding MC³. See "Configuring the MC³" on page 6.

For each node, you insert two transactions, one for reading data from the MC³, and one for writing data to the MC³. The MC³ controllers support two transaction types, "Read Holding Registers" (Modbus Function 3) and "Preset Multiple Regs" (Modbus Function 16). You use "Read Holding Registers" to retrieve data from the MC³'s and "Preset Multiple Registers" to write data to the MC³'s

Each transaction has parameters for a "Query" and a "Response". They all have to be set correctly. Most parameters can be summarily set according to the table below, but the positioning of the data in the EIP data buffer table is critical and somewhat complex. The configuration used here contains what you would need to monitor and control the MC³ controllers. Referring to the MC³ - CIT Format on page 10, you read from words 16 - 42 and write to words 44 - 51. This will allow you maximum monitoring capabilities, and to set control bits, external inputs, the feedrate setpoint, the batch setpoint (when used) and one multiplexable MC³ parameter.



The Sub Network related parameters should be set according to the following table. The settings are such that there is reasonable performance degradation for four connected controllers, meaning that the communication turn-around time is in the same order of magnitude as the internal MC³ update time.

Parameter	Setting	Comment
Offline options for Fieldbus	Freeze, Clear or Stop Scanning.	Determines what happens to the write transactions on the Modbus side if EtherNet/IP goes down. Stop Scanning makes it impossible to check communications between MC³ and EIP without a working EtherNet/IP connection. Use Freeze.
Offline options for sub-network	Clear or Freeze	Determines what happens to the read transactions on the EtherNet/IP side if Modbus communications goes down. Freeze is an attractive solution since it does not disrupt data from all controllers if just one goes offline. There are better means to monitor network integrity. See "Use the integrity bit." on page 22.
Update mode	Cyclically	There is no need to do triggered updates. Performance is good enough with cyclical update.
Minimum time between Broadcasts	10	Broadcasts are not used. This setting has no meaning.
Reconnect time	25	Trying to re-connect too often will degrade performance. 250 ms is a reasonable value
Retries	3	It appears that retries are not done consecutively. The performance penalty for a retry is minimal.
Timeout time	25	The MC³ normally responds within 10 ms, but if a "heavy" screen, such as the line graph screen is displayed on the controller, the response time can be as long as 200 ms.
Update time	1	Set to 10 ms to maximize throughput.
Trigger Address	0x05FF	Not used. This is the default.

Use the following tables to set the parameters in all transactions:

Read Holding Registers Query

Parameter	**MC³ #1**	**MC³ #2**	**MC³ #3**	**MC³ #4**
Slave Address	0x01	0x02	0x03	0x04
Function	0x03	0x03	0x03	0x03
Starting Address (Hex Word)	0x0010	0x0010	0x0010	0x0010
Number of points (Hex Words)	0x001C	0x001C	0x001C	0x001C
Checksum type, Start Byte	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000

Read Holding Registers Response

Parameter	**MC³ #1**	**MC³ #2**	**MC³ #3**	**MC³ #4**
Slave Address	0x01	0x02	0x03	0x04
Function	0x03	0x03	0x03	0x03
Byte Count (Hex. Always twice the Number of points)	0x0038	0x0038	0x0038	0x0038
Data length,	0x0038	0x0038	0x0038	0x0038
Data location	0x0004	0x003C	0x0074	0x00AC
Byte Swap	Swap 2 bytes	Swap 2 bytes	Swap 2 bytes	Swap 2 bytes
Checksum type, Start Byte	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000

Preset Multiple Registers Query

Parameter	**MC³ #1**	**MC³ #2**	**MC³ #3**	**MC³ #4**
Slave Address (Hex)	0x01	0x02	0x03	0x04
Function (Hex)	0x10	0x10	0x10	0x10
Starting Address (Hex Word)	0x002C	0x002C	0x002C	0x002C
Number of Registers (Hex Words)	0x0008	0x0008	0x0008	0x0008
Byte Count (Hex. Always twice the Number of points)	0x10	0x10	0x10	0x10
Data length (Hex bytes)	0x0010	0x0010	0x0010	0x0010
Data location in the EIP table	0x0204	0x0214	0x224	0x234
Byte swap	Swap 2 bytes	Swap 2 bytes	Swap 2 bytes	Swap 2 bytes
Checksum type, Start Byte	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000

Preset Multiple Registers Response

Parameter	**MC³ #1**	**MC³ #2**	**MC³ #3**	**MC³ #4**
Slave Address	0x01	0x01	0x03	0x04
Function	0x10	0x10	0x10	0x10
Starting Address (Hex Word)	0x002C	0x002C	0x002C	0x002C
Number of Registers (Hex Words)	0x0008	0x0008	0x0008	0x0008
Checksum type, Start Byte	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000	CRC, 0x0000

Don't forget to download the configuration to the ABC-PDP and to save the configuration file for future reference.

CONFIGURING THE MC³

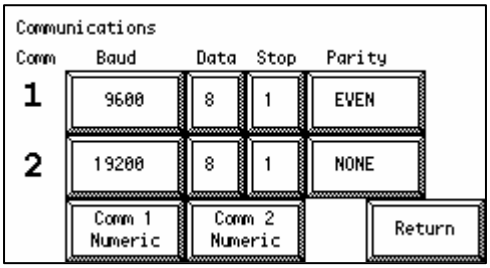
Configuring the MC³ controllers include

- Setting up communications parameters
- Setting up register tags, if you want to monitor special parameters, other than Feedrate, Belt Load, Belt Speed, Subtotal or Total.
- Configuring Warnings and Faults. This configuration is done regardless if communications is used or not.
- Mapping external inputs and outputs. Any logical input that you want to control from the PLC has to be mapped to an external input. Any logical output you want to monitor has to be mapped to a logical output.
- Setting up the Setpoint Source so that the Setpoint is taken from the PLC.

The following menu references and screen shots were taken using the MC³ 20.20.EX.A Belt Feeder Controller application.

The "MC³ 20.20.EX Weigh Feeder Controller Operation and Maintenance Manual, Version A" (O&M) is available at the Merrick Web Site: http://www.merrick-inc.com/mct/, at the bottom of the page, click on MC³ Firmware Application Overview.

Setting the MC³ communications parameters



To get to the "Communications" screen from the main screen, touch **Action Menu**, **Settings Menu**, enter the password, **Inputs & Outputs** and finally **Comm Settings**. Modbus RTU runs on COM 1. In this example, the standard RTU line parameters were selected, 9600 baud, 8 data bits, 1 stop bit and Even parity. These parameters must agree with the settings in the ABC-PDP "Sub-Network" settings on page 4. Refer to O&M, page 66 for more details.

Touching the **Comm 1 Numeric** button takes you to the **Comm Numeric Params** screen. Set the parameters as follows:

Parameter	Value	Comment
Controller #	1 to 4	This is the Slave Address. They must be different for each controller and correspond to the "Slave Address (hex)" parameters in the query and response parameter list in the ABC-PDP configuration
Start Char	10	Has no meaning for Modbus RTU
End Char	13	Has no meaning for Modbus RTU
Comm Timeout	5.0	The MC³ times incoming, legal RTU telegrams, and turns on the **Ser Comm Lost** logical output if this time expires. You typically let this output qualify a Warning. Also, this will turn off all External Inputs.
Comm Protocol	2	Enter 2 for Modbus RTU
Write Protect	3071	BFF hex. All registers write protected except Primary Setpoint
Word Order	0	All registers have normal word order. This has to be found by trial and error. The ControlLogix PLC has the word order for a floating point value straight (as compared with a PLC-5 or SLC-5).
Int/Frac FP	0	Floating Point transfer is supported. Integer/Fraction is not needed.
Tag Reg 1	353	Internal MC³ Register Number for the "Zero Tracking Load" variable. This will make the value show up in the "Tag 1 R value" position.

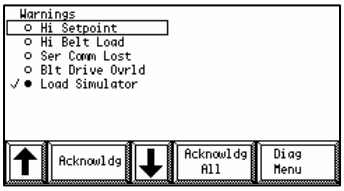
Parameter	Value	Comment
Tag Reg 2	0	Use 0 for unused tags
Tag Reg 2	0	
Tag Reg 3	0	
Tag Reg 4	0	
Tag Reg 5	0	

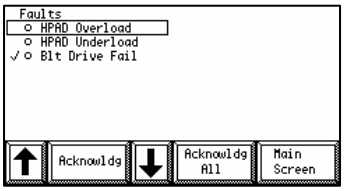
Obviously, to be able to tag other internal MC³ registers, you need to have the register specification for the specific MC³ application you are using. You can find the register specifications for most MC³ firmware applications at the Merrick Web Site, http://www.merrick-inc.com/mct/, at the bottom of the page, click on MC³ Firmware Application Overview.

Configuring warnings and faults

Warnings and Faults are qualifiers to logical inputs and outputs, normally set by the user. Warnings are considered to require attention. Faults are considered to be fatal for the feeder operation, and the controller will attempt to stop the feeder. See O&M, page 55. In this example the warnings and faults are set up according to the following table:

Logical I/O	Qualifier	Comment
HPAD Overload	Fault	Invalid Load Cell Signal
HPAD Underload	Fault	Invalid Load Cell Signal
Blt Drive Fail	Fault	Signal from the belt motor VFD, connected to an MC³ input.
Hi Setpoint	Warning	Setpoint above upper limit. To detect floating point transfer problems.
Hi Belt Load	Warning	Too much material on the belt
Ser Comm Lost	Warning	No valid RTU telegram has been received for 5 seconds.
Load Simulator	Warning	Don't run the feeder for real with the Load Simulator turned on!
Blt Drive Ovrld	Warning	Signal from the belt motor drive, connected to an MC³ input.





The qualifiers are set up in the Digital Inputs (O&M Page 55) and Digital Outputs (O&M Page 59) screens. With the settings above, the Warnings and Faults screens look like this.

The state of the checkmark is transferred to the Warnings [18] and Faults [19] word in the CIT. The bit order is the same as the displayed order of the screen. It is important to note that the bits in the Warnings and Faults registers reflect the state of the checkmark, not the dot. In this warning screen, both are on for the logical input "Load Simulator". Bit 3 of the Warnings register is on. If the "Load Simulator" input is turned off, then the dot goes away, but the checkmark stays until the warning is ACK'd, either on this screen or by the "Clear Warnings Command" bit in the Control [44] register.

Note that the order of display is not configurable. It is derived from the order of logical outputs and logical inputs in the Digital Inputs and Digital Outputs screens. If you add or remove a warning or fault qualifier to a logical input or output, the order may change.

Configuring External Inputs and Outputs

Logical inputs and outputs can be mapped in three ways:

1. To a physical input or output. In this example the logical input 'Belt Drive Overload' is mapped to Rack 1 Input 3, which, in turn, is connected to the Overload output of the belt motor VFD. The physical output Rack 1 Output 5 is mapped to the logical output 'Drive Enable'. The output is then connected to the Start input of the belt motor VFD.
2. To an external input or output. In this example, the 'Run Permission' logical input is mapped to External Input 1. This allows the PLC to start and stop the feeder through the External Inputs register, CIT Word 45, bit 0.
3. Unused Logical Inputs are typically connected to the Physical Inputs 'Always On' or 'Always Off'.

The PLC controls inputs to the MC³ as bits in the 'External Inputs' register, CIT Word 45. They are then mapped to Logical Inputs in the MC³. Note that the External Inputs are numbered 1 - 16. Bits in the 'External Inputs' word are numbered 0 - 15 in the PLC.

In this example, we use 5 inputs. Two are physical connections from the VFD to the MC³, one is a physical connection to the emergency stop circuit (Feeder Block), and two are inputs controlled from the PLC (Run Permission, Load Simulator).

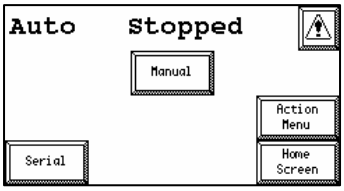
This is how the inputs have to be mapped in the MC³, where n is 1 - 4 for MC #1 - #4:

Logical	Physical	W/F	Bits in the PLC
Run Permission	External Input 1		MC3[n].Ctls[1].0
Load Simulator	External Input 2	W	MC3[n].Ctls[1].1
Feeder Block	Rack 1 Input 1		
Belt Drive Ovrld	Rack 1 Input 2	W	
Belt Drive Fail	Rack 1 Input 3	F	

Digital output mapping:

Physical	Logical	W/F	Bits in the PLC
Rack 1 Output 4	Fdr Drv Enable		
External Output 1	Fault		MC3[n].Status[1].0
External Output 2	Warning		MC3[n].Status[1].1
External Output 3	Ready		MC3[n].Status[1].2
External Output 4	Good Feedrate		MC3[n].Status[1].3
External Output 5	Hi Belt Load	W	MC3[n].Status[1].4
External Output 6	Hi Setpt	W	MC3[n].Status[1].5
External Output 7	HPAD Overload	F	MC3[n].Status[1].6
External Output 8	HPAD Underload	F	MC3[n].Status[1].7

Setting up the setpoint source



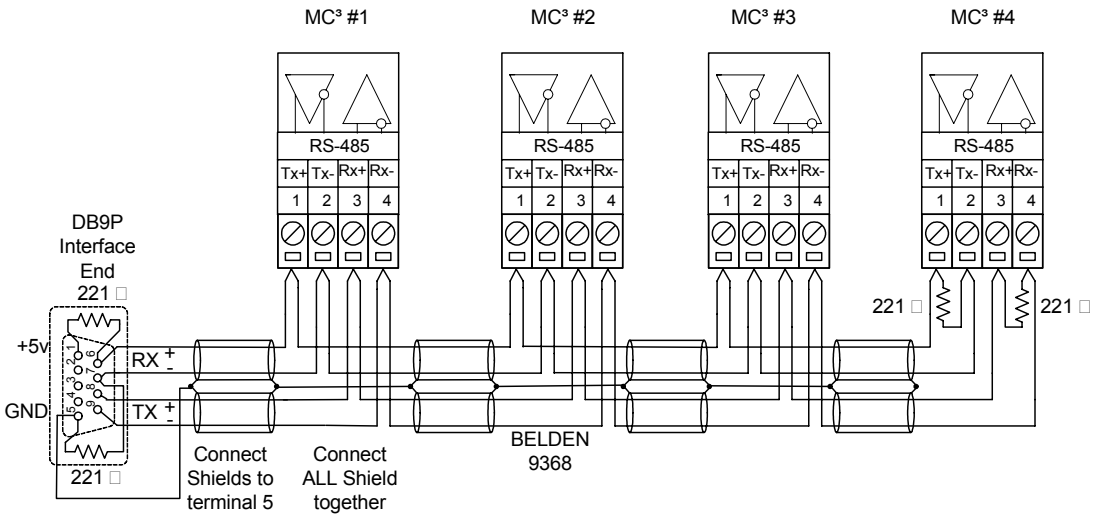
The MC³ Setpoint Method must be set to Serial. See O&M, Page 27. The setpoint is taken from the PLC's MC3[0].CVars[0] in this example.

CONNECT THE EIP TO THE MC³S AND CHECK COMMUNICATIONS

If the distance between the EIP and the MC³ controllers is more then a few feet, use a cable designed for RS-422. There should be two pairs, individually shielded. Belden 9368 or equivalent is a good alternative. Connect the shields only at the ABC-PDP, to pin 5 of the DB9-P. At the MC³s, connect the incoming and outgoing shields together only. Add pull-up and pull down

resistors at the EIP, 221 Ω, between pins 6 and 1 (Rx+ and +5V) and pins 7 and 5 (Rx- and GND). Add terminating resistors at the last MC³, 221 Ω. Connect one between terminals 1 and 2 and one between terminals 3 and 4. The pin numbering on the MC³ is left to right.

There are two revisions of the MC³ controller, "old" and "new" Refer the MC³ hardware manual. It is available at the Merrick Web Site, http://www.merrick-inc.com/mct/, at the bottom of the page, click on MC³ Hardware Reference Manual. The different layouts are described in "MC³ Connections".



A "New" CPU board, Rev 11 or later, have two RS-485 ports built in. You connect to the right port on the bottom board. The DB9-P (Pins) connector to the right is the RS232 port for Comm 1. The two connectors on the left are for Comm 2.

An "Old" CPU board did not have an RS-485 port, so the port on the top board was used. For a card stack with a "New" CPU board, this port is not connected and does not work.

When you have connected and powered up the MC³s and ABC-PDP, communication should start even without any EtherNet/IP connection. Check that all looks OK in the ModBus Diagnostic screen. See page 20.

Bench-Testing Your System

If you have access to the MC³'s and the EIP, you can bench-test your entire system, by using the built-in simulators in the MC³. The MC³'s will behave as if a real feeder was connected to them, although you will have to set up a belt load manually. To put an MC³ 2020 in simulation mode:

1. In **Action Menu, Settings Menu, Tacho Settings**, push the lower left button until it reads "**Simulator**"
2. In **Action Menu, Settings Menu, Inputs & Outputs, HPAD settings**, push the leftmost button until it reads "**None**"
3. Set the Load Simulator bit, MC3[n].Ctls[1].1 in the CLX. You control the belt load by invisible touch pads on the upper row on the MC³ screen. Increase the load with touch pad 6 (from the left). Decrease the belt load with touch pad 5.

MC³ - CIT FORMAT

The format of the data structure exposed by the MC³ to communications is the same for Modbus ASCII, Modbus RTU and DF-1 CIF read and write. They are also common for all MC³ firmware applications. The first 16 words are in place for legacy reasons and should not be used. However, reading and displaying words 8 - 15 can sometimes be useful for troubleshooting purposes.

Words 16 - 43 are Read Only and intended for monitoring. For performance reasons, read a section, starting with word 16, and as far along as you need. The performance penalty for extending the read area length is small; however, you may find that you run out of space in the EIP's buffer if you have many MC³'s.

If you need to conserve EIP buffer space, first determine how many variables you really need. Typically, for a belt feeder, this would be Feedrate and Subtotal. If this is the case, you can read from word 16 – 23. You will then have to enter the MC³ register number for Feedrate in Tag 1 RegNo (251 for 20.20.EX.A) and for Subtotal in Tag 2 RegNo (181 for 20.20.EX.A). The Tag X RegNo values can be set in the MC³ by going to Action Menu, Settings Menu, enter the password, Inputs & Outputs, Comm Settings, Comm 1 Numeric.

Words 44 - 67 can be written to. Note that the "Tag n W value" (n = 1...5) refers to the same internal MC³ register as the "Tag n R value". The "Tag n RegNo" is the internal MC³ register number for Tag n. This Tagging scheme is in place to enable you to get to any variable within the MC³. You must have the Register Specification for the application and version you are working with. You can find the register specifications at the Merrick Web Site, http://www.merrick-inc.com/mct/, at the bottom of the page, click on MC³ Firmware Application Overview.

There is a risk here. You can easily crash the MC³ if you, for example, write a value of zero to a variable used in a divide operation. The only remedy in this case is to Ram Reset the MC³. You will then have to enter all the MC³ parameters again. If you have WinMerik, which can be used to upload and download all MC³ parameters with a PC; it is a good idea to upload before you start writing to internal MC³ variables.

Words 60 to 67 are also accessible in the MC³ Comm Numeric Params screen. See "Setting the MC³ communications parameters" on page 8. If you write to them from the ABC-PDP, the values set in that screen will be overwritten. Words 60 to 67 are "sticky" and will survive power cycling.

Word formatting and write protection properties are set in words 60 - 62. It is a good idea to have all tagged registers write protected (4095, 0xFFF in word 60) until the floating point transfer method has been checked out. It is possible to transfer an invalid floating point number into a variable that is used in actual calculations, which will also crash the MC³.

The following table lists all transitions from the MC³ CIT table to the data structures in the PLC in this example. Note that this is example specific.

CIT Word	Ln	Contains	Data format Bit (Note 2)	CIT table Element	Comment	Structure member in MC3[n] in CLX. n = 1…4
0	8	Physical I/O			Backwards Compatibility	
8	8	Logical I/O			Backwards Compatibility	
16	1	Status		R 0	See Note 1 below for bit info	MC3[n].Status[0]
17	1	External Outputs		R 1	Bit 0: Ex Out 1, Bit 1: Ex Out 2 etc	MC3[n].Status[1]
18	1	Warnings		R 2	As they appear in the MC³ Warnings screen	MC3[n].Status[2]
19	1	Faults		R 3	As they appear in the MC³ Faults screen	MC3[n].Status[3]
20	2	Tag 1 R Value	0 (0001)	R 4,5	Register pointed to by Tag 1 RegNo	MC3[n].SVars[0]
22	2	Tag 2 R Value	1 (0002)	R 6,7	Register pointed to by Tag 2 RegNo	MC3[n].SVars[1]
24	2	Tag 3 R Value	2 (0004)	R 8,9	Register pointed to by Tag 3 RegNo	MC3[n].SVars[2]
26	2	Tag 4 R Value	3 (0008)	R 10,11	Register pointed to by Tag 4 RegNo	MC3[n].SVars[3]
28	2	Tag 5 R Value	4 (0010)	R 12,13	Register pointed to by Tag 5 RegNo	MC3[n].SVars[4]
30	2	Feedrate	5 (0020)	R 14,15	As seen in the main screen	MC3[n].SVars[5]
32	2	Weight/Load	6 (0040)	R 16,17	As seen in the main screen	MC3[n].SVars[6]
34	2	Speed Info	7 (0080)	R 18,19	Speed, if available, or Drive CV	MC3[n].SVars[7]
36	2	Subtotal	8 (0100)	R 21,22	As seen in the main screen	MC3[n].SVars[8]
38	2	Total	9 (0200)	R 23,24	As seen in the main screen	MC3[n].SVars[9]
40	1	App/Ver		R 24	App # in hi byte, Ver (ASCII) in low	MC3[n].Status[4]
41	2	Phys Inputs		R 25	Note 7	MC3[n].Status[5]
42	1	Phys Outputs		R 26,27	Note 8	MC3[n].Status[6,7]
44	1	Control		W 0	See Note 1 below for bit info	MC3[n].Ctls[0]
45	1	External Inputs		W 1	Bit 0: Ex In 1, Bit 1: Ex In 2 etc	MC3[n].Ctls[1]
46	2	Primary Setpoint	10 (0400)	W 2,3	Typically Feedrate Setpoint	MC3[n].CVars[0]
48	2	Sec. Setpoint	11 (0800)		Typically Batch Setpoint	MC3[n].CVars[1]
50	2	Tag 1 W Value	0 (0001)		Register pointed t by Tag 1 RegNo, for writing	MC3[n].CVars[2]
52	2	Tag 2 W Value	1 (0002)			Not written to in this example

CIT Word	Ln	Contains	Data format Bit (Note 2)	CIT table Element	Comment	Structure member in MC3[n] in CLX. n = 1…4
54	2	Tag 3 W Value	2 (0004)			
56	2	Tag 4 W Value	3 (0008)			
58	2	Tag 5 W Value	4 (0010)			
60	1	Write Protect Bits			Set for write protection. See note 3	
61	1	Word Order Bits			Set to reverse. See note 4	
62	1	Int/Frac Bits			Set for Int/Frac. See note 5	
63	1	Tag 1 RegNo			MC³ register number	
64	1	Tag 2 RegNo				
65	1	Tag 3 RegNo				
66	1	Tag 4 RegNo				
67	1	Tag 5 RegNo				

Note 1 This is the bit assignment for words 16 and 44, corresponding to the first words read and written by the ABC-PDP.

Bit	Word 16 Function	Word 44 Function	Comment
0-6	See note 6	Unused	Tag error reporting
7	Integrity bit echo	Integrity bit	The MC³ will echo this bit from word 44 to word 16.
8	Clear Warnings Done	Clear Warnings Demand	Used by PLC to clear all warnings. Set the bit in word 44, and wait for the bit in word 16 to set. Then clear the bit in word 44.
9	Clear Subtotal Done	Clear Subtotal Demand	Same scheme as for Clear Warnings
10	Lock Touchpad Done	Lock Touchpad Demand	Disables all touch-buttons on the MC³ A Ser Comm Lost will re-enable them.
11	Reserved	Reserved	Planned for Register Download
12	Pacing flag		Low Feedrate Deviation. Used for pacing functions, whereby other feeders in the system will follow a "starving" feeder.
13	Not Serial Setpoint		MC³ will ignore sent setpoint. Set when the Setpoint Method is something else than Serial.
14	MC³ in menu		Used for tampering monitoring. This bit is on whenever the MC³ menu system is entered.
15	MC³ recalibrated	Recalibration ACK	In place for historical reasons. Set when any calibration procedure is accepted or any parameter is changed. Reset with a Low-to-high transition of Recalibration ACK.

Note 2. This column defines the bit weight for the corresponding variable in the format words 60, 61 and 62. Example: Tag 2 Read and Write values both are governed by bit one (with bit weight 0002 Hex) in the format words 60, 61 and 62. In this way, Write Protection, Word Order and Integer/Fraction representation is individually settable for every numerical variable in the data map.

Note 3. The write protection property should be set when an MC³ register is tagged for monitoring only. When writing to words that are write protected, the corresponding Tag n W value changes accordingly, but the corresponding MC³ register (which you see in Tag n R Value) is unaffected. This is useful for testing data transfers to the MC³ before they are implemented, or when you need to change a variable only at certain instances.

Note 4. The Word Order Bit, when set, reverses the order of the two words that contains value information. To correctly transfer floating-point values to and from PLC's, these bits may have to be set.

Note 5. The Integer/Fraction bits are used when the device using the data does not support floating-point numbers. With the corresponding Word Order bit cleared, the first word will carry the Integer part, and the second the fractional part, multiplied with 10000. (4 implied decimal places). Note that for a negative value, both the integer and fractional

parts are negative. This scheme will not work for values outside the interval (-32767..32767), since those are the extreme values for an INT.

Note 6. Bits 0 - 3 are used to signal problems with Tag Register Numbers as follows:

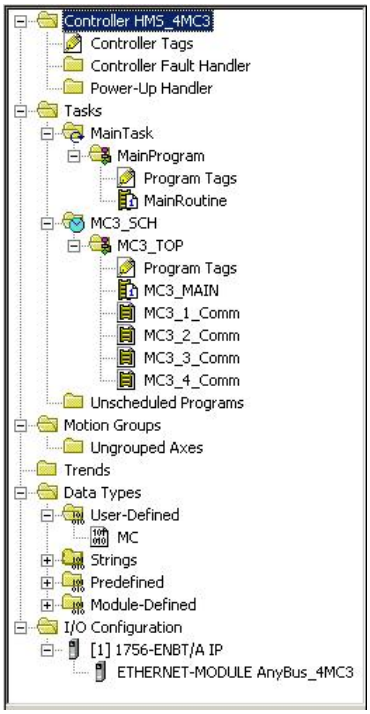
Bit 0 Attempt to write to a conditionally write-protected MC³ register while the "Extended Access" logical input is OFF.
Bit 1 Attempt to write to a write-protected register.
Bit 2 Attempt to write to a non-existing register.
Bit 3 Attempt to read from a non-existing register. Zero returned.

Note 7 Rack 1 input 1 in bit 0, Rack 1 input 2 in bit 1 etc. Rack 2 input 1 in bit 4….

Note 8 Rack 1 output 1 in 42 bit 0… Rack 1 output 8 in 42 bit 7, Rack 2 output 1 42 bit 8…Rack 2 output 2 in 42 bit 15, Rack 3 output 1 in 43 bit 0 etc.

CONFIGURING CONTROLLOGIX PLC

The RSLogix5000 (V10.00.00) file for this example can be downloaded from the Merrick connectivity web site.



To verify CLX to MC³ communication integrity, a test set-up was arranged. A user defined data type, MC, was defined. A four element array of that data type was allocated, with tag name MC3. The MC data type resembles the CIT. The EIP was configured as an EtherNet/IP adapter. Some logic was added to convert the raw data in the CLX adapter I/O tables into MC3[], and to monitor the communications integrity.

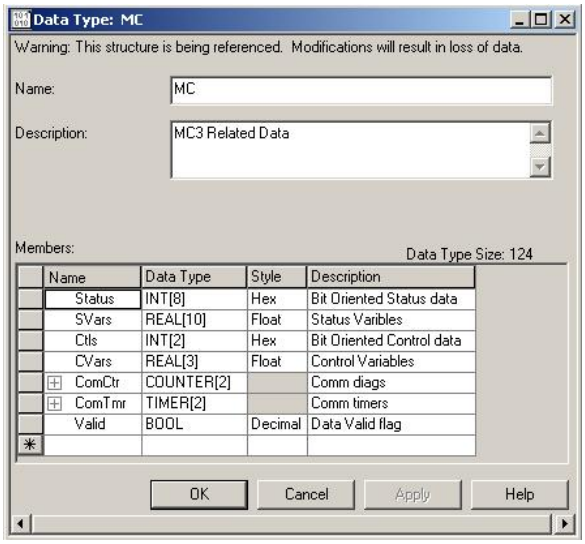
The communications related logic is running as a scheduled task, with 500 ms interval. There is no reason to run this task more often, since the turn-around time on the Modbus side is around 800 ms. The priority has to be at least higher (lower number) than the Main Task (10).

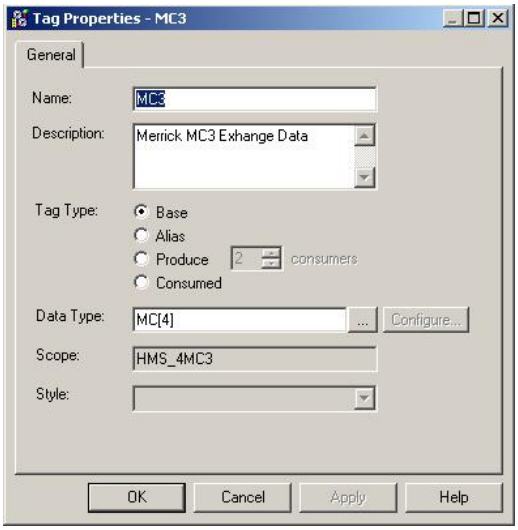
MC3_MAIN calls the four MC3_n_Comm routines, one for each MC³ controller. It is possible to use only one such routine, and call it four times with parameters. This saves controller memory, but makes it harder to debug the system.

Several versions of the test logic used can be found in the Merrick communications website. It is possible to copy and paste between different RSLogix5000 projects. To incorporate the definitions and logic from the test file into your project, you must first convert our project to your version. The easiest way to do this is to import the L5K file into a new project, and then cut and paste objects from the new project into your existing project. Proceed in the following order:

1. I/O-configure the EIP, using the name "AnyBus_4MC3" in I/O configuration. If you use another name, you have to edit all ladder files. You can rename the adapter later.
2. Copy and paste the user defined data type MC.
3. Copy and paste the tag MC3 in Controller Tags.
4. Copy and paste MC3_SCH in Tasks.
5. Copy and paste MC3_TOP in MC3_SCH. This will take Program Tags, MC3_MAIN and all four MC3_n_COMM routines with it.

MC User Defined Data type





The MC User Defined Data Type defines the layout of the elements in MC3[n], which is your interface to MC³ data. After creating it, you can add a tag of an array of MC. The layout resembles the CIT table, but the data types are choosen to make it easy to write ladder logic to monitor and control the MC³'s.

Status CIT elements 16 - 19, 40 – 27. Status bit information from MC³'s to PLC.

Svars CIT elements 20 – 39. Variable information from MC³'s to PLC.

Ctls CIT elements 44 - 45. Control bit information from PLC to MC³'s.

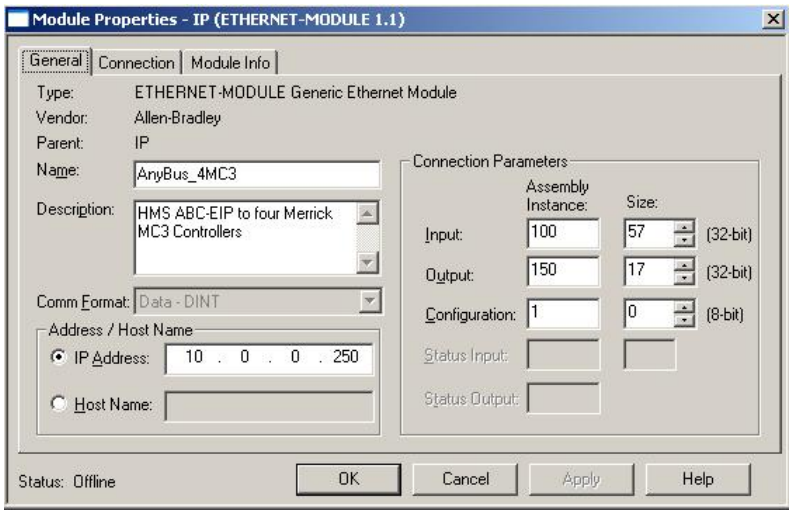
Cvars CIT elements 46 – 51. Variable information from PLC to MC³'s.

ComCtr will count successful and failed communications go-arounds.

ComTmr are internally used timers to monitor communications integrity

Valid True when communications is running without errors.

ControlLogix I/O Configuration



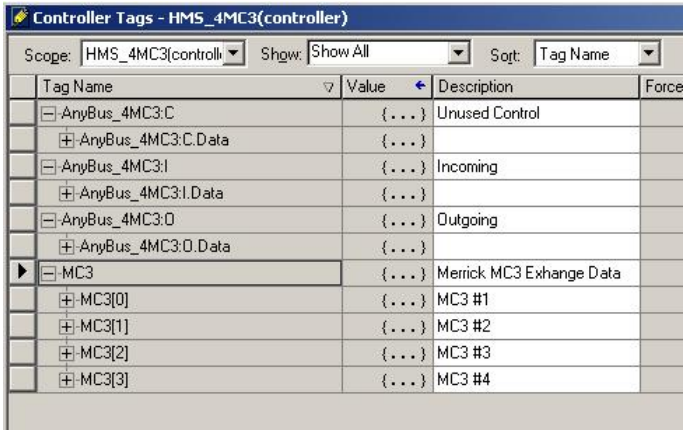
To configure the EIP as a I/O module under the PLC Ethernet adapter, you must first install the EDS file. The file itself is available at the HMS web site, http://www.anybus.com. Use the Rockwell Software EDS Hardware Installation Tool. It is available in the RSLinx start menu by default. The configuration is described in detail in the HMS Application Note "Establishing I/O communication between AnyBys-S EtherNet/IP and

ControlLogix5000", available on the HMS web site. As you configure the module, use DINTs for the Comm Format field. This is how the Module Properties General tab look in this example. You can calculate the sizes for n MC³ controllers

Input Size = 1 + 14n

Output Size = 1 + 4n

Use a reasonable Requested Packet Interval in the Connection tab. There is no reason the use the default (5 mS). It would cause unnecessary network traffic. The turn-around time is about 800 ms inside the EIP. We used 100 ms in this example.



When you have completed the configuration, you will find three brand new tags in the controller scope. The tag names are derived from the name we gave the EIP at configuration time.

AnyBus_4MC3:I.Data is an array of 57 DINTS, that holds the entire input buffer in the EIP.

AnyBus_4MC3:O.Data is an array of 17 DINTS, holds hold the entire output buffer in the EIP.

WARNING Configuring the EIP as an I/O device will generate UDP traffic on your EtherNet/IP network. If you use network switches to segment your traffic, you will find that most switches will broadcast all the messages on all ports. You can easily use up all your network bandwidth this way. There is an alternative way. Don't configure the EIP as an I/O device at all, and write CIP generic messages instead. That will generate TCP traffic rather than UDP, and the switches will behave as you expect them to. There is a HMS publication that describes this: "Reading/writing data from AnyBus-S EtherNet/IP using ControlLogix5000 MSG instruction", available from the HMS website.

PLC logic to move the data

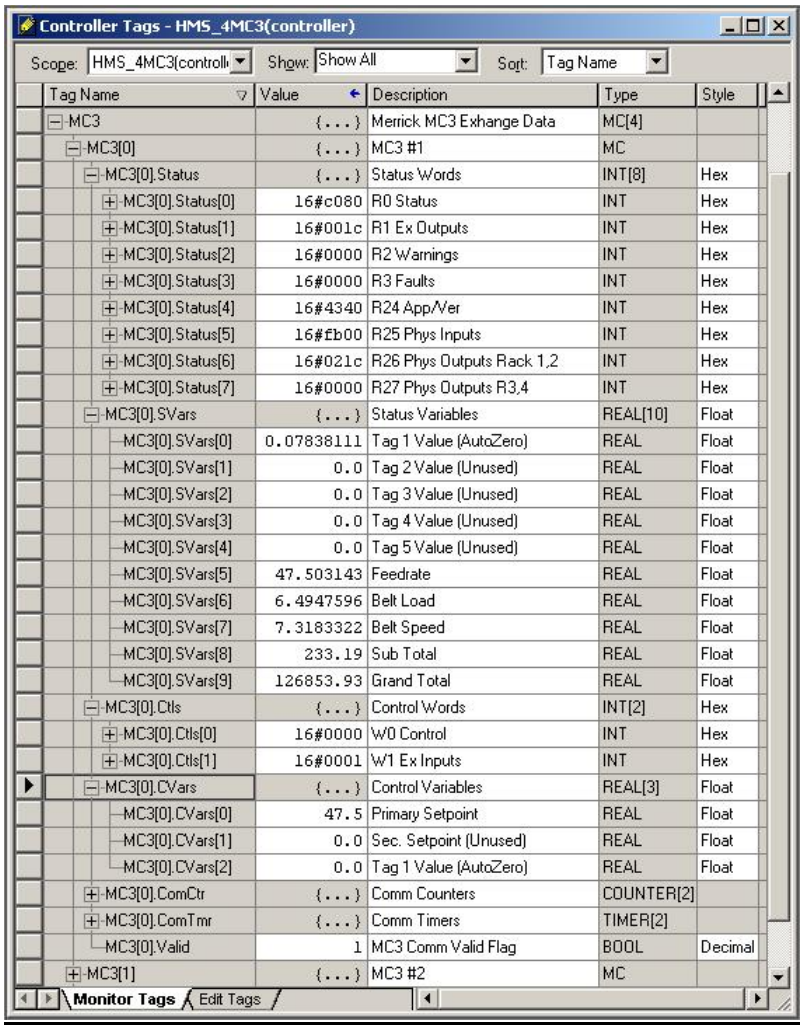
The AnyBus_4MC3 tags are not suitable for ladder logic programming. For that reason, some data-shuffling logic was added to move data between the AnyBus tags and the MC3 tag. With this scheme there is a direct relation between the tag monitor in the CLX and the Modbus data diagnostic screen in the MC³.

MC3_MAIN holds off ladder execution for 20 seconds at boot, sets the Data Valid bit for the Fieldbus side of the EIP, calls the four communication routines for the four MC³'s and maintains the handshake confirmation bit for the EIP.

MC3_n_Comm, where n = 1..4, are the individual communication handling routines for the four MC³'s. It copies data between the AnyBus_4MC3 tags and the MC3[n] tag, toggles the integrity bit, maintains communications statistics and verifies that the communication is healthy.

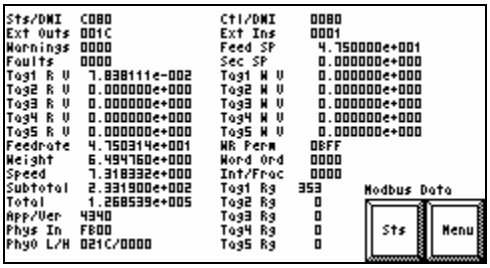
Ladder listing files are available at the Merrick website.

Here is a screen shot monitoring the MC3[0] tag elements.



Tag Name	Value	Description	Type	Style
MC3	{...}	Merrick MC3 Exhange Data	MC[4]	
MC3[0]	{...}	MC3 #1	MC	
MC3[0].Status	{...}	Status Words	INT[8]	Hex
MC3[0].Status[0]	16#c080	R0 Status	INT	Hex
MC3[0].Status[1]	16#001c	R1 Ex Outputs	INT	Hex
MC3[0].Status[2]	16#0000	R2 Warnings	INT	Hex
MC3[0].Status[3]	16#0000	R3 Faults	INT	Hex
MC3[0].Status[4]	16#4340	R24 App/Ver	INT	Hex
MC3[0].Status[5]	16#fb00	R25 Phys Inputs	INT	Hex
MC3[0].Status[6]	16#021c	R26 Phys Outputs Rack 1,2	INT	Hex
MC3[0].Status[7]	16#0000	R27 Phys Outputs R3,4	INT	Hex
MC3[0].SVars	{...}	Status Variables	REAL[10]	Float
MC3[0].SVars[0]	0.07838111	Tag 1 Value (AutoZero)	REAL	Float
MC3[0].SVars[1]	0.0	Tag 2 Value (Unused)	REAL	Float
MC3[0].SVars[2]	0.0	Tag 3 Value (Unused)	REAL	Float
MC3[0].SVars[3]	0.0	Tag 4 Value (Unused)	REAL	Float
MC3[0].SVars[4]	0.0	Tag 5 Value (Unused)	REAL	Float
MC3[0].SVars[5]	47.503143	Feedrate	REAL	Float
MC3[0].SVars[6]	6.4947596	Belt Load	REAL	Float
MC3[0].SVars[7]	7.3183322	Belt Speed	REAL	Float
MC3[0].SVars[8]	233.19	Sub Total	REAL	Float
MC3[0].SVars[9]	126853.93	Grand Total	REAL	Float
MC3[0].Ctls	{...}	Control Words	INT[2]	Hex
MC3[0].Ctls[0]	16#0000	W0 Control	INT	Hex
MC3[0].Ctls[1]	16#0001	W1 Ex Inputs	INT	Hex
MC3[0].CVars	{...}	Control Variables	REAL[3]	Float
MC3[0].CVars[0]	47.5	Primary Setpoint	REAL	Float
MC3[0].CVars[1]	0.0	Sec. Setpoint (Unused)	REAL	Float
MC3[0].CVars[2]	0.0	Tag 1 Value (AutoZero)	REAL	Float
MC3[0].ComCtr	{...}	Comm Counters	COUNTER[2]	
MC3[0].ComTmr	{...}	Comm Timers	TIMER[2]	
MC3[0].Valid	1	MC3 Comm Valid Flag	BOOL	Decimal
MC3[1]	{...}	MC3 #2	MC	

Here is the corresponding screen shot from the MC³ Modbus Data Diagnostic screen.



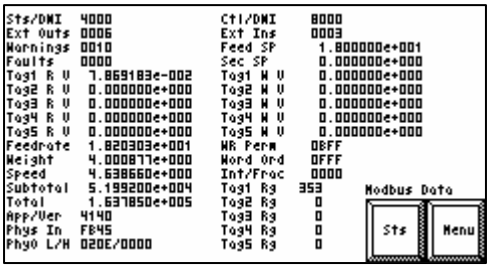
Troubleshooting

Setting up industrial networks can sometimes be a daunting task. In this example, four mapping layers are involved, along with two different communication protocols. Fortunately, there are excellent troubleshooting tools available.

Look at the LED's on the MC³ CPU board.

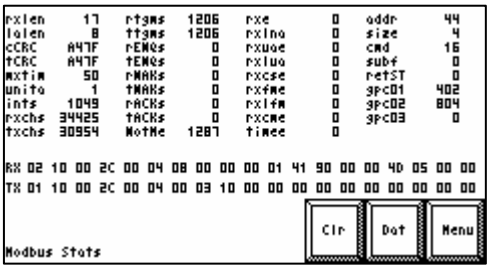
MC³ CPU boards Rev 1 and later have a set of LTD indicators. Indicators 3 and 4 are connected to Comm 1 serial Receive Data and Transmit Data, respectively. Indicator 3 should blink four times as fast as indicator 4, since three out of four telegrams are intended for other controllers in this example. The rightmost indicator in this picture is indicator 1.

Check data in the CIT data screen.



The actual values in the CIT exposed to communications can be inspected in the MC³ by touching Action Menu, Diag Display, Modbus Diag, Dat. Note that the values are only updated on valid Modbus telegrams. If no telegrams have been received, most values are zero. As you can see, the layout follows the CIT exactly. All integer values are presented in hexadecimal format except the Tag register numbers. The 'e' format for the floating points can help troubleshooting FP transfers. You are reading from all rows to the left, and writing to the first five rows on the right in this example. If you succeed with the integrity bit, you should see bit 7 in the Sts/DNI and Ctl/DNI toggle.

Check error counters in the Communication Diagnostic screen.



Communications status and statistics can be inspected in the MC³, by touching Action Menu, Diag Display, Modbus Diag. The screen looks like this.

In this shot, out of 1206 successful exchanges, there was none lost to errors. There were 1287 telegrams addressed to some other RTU device. The last command received was 16 (Preset Multiple Regs), starting at word 44, 4 words long.

Label	Meaning
rxlen	Length, in bytes, of the last incoming RTU telegram
lalen	Length, in bytes, of the last outgoing RTU telegram
cCRC	CRC16 value calculated out of the incoming telegram. Hex.

Label	Meaning
tCRC	CRC16 value received in the incoming telegram. Hex. Should be the same as cCRC
mxtim	Communications timeout in 100 ms ticks – “Comm Timeout” in the Com/Num menu
unita	“Slave address” for this controller number. “Controller Number” in Com/Num menu
ints	Comm events counter. Counts all incoming and outgoing bytes
rxchs	Received bytes counter.
txchs	Transmitted bytes counter
rtgms	Received, complete telegrams to this slave counter
ttgms	Transmitted telegrams from this slave counter
rENQs	Not used for Modbus.
tENQs	Not used for Modbus
rNAKs	Not used for Modbus
tNAKs	Transmitted NAK counter. Telegrams to this node with badly formatted data, requesting non-existing registers or writing to write-protected registers
rACKs	Not used in Modbus
tACKs	Not used in Modbus
NotMe	Counter for Received telegrams intended for other slaves
rxe	Received telegrams in error counter
rxlna	Latest NAK. See note 2.
rxuae	Received bytes with UART errors counter
rxlua	Last encountered UART error. See note 1.
rxcse	Received telegrams with CRC16 error counter
rxfme	Received telegrams with format error counter
rxlfm	Last format error encountered. See note 2.
rxcme	Non-supported command received counter
timee	Comm timeouts counter
addr	First register in received command. Should toggle between 16 and 44
size	Number of registers in received command. Should toggle between 28 and 4
cmd	Modbus command received. Should toggle between 3 and 16
subf	Subfunction in diagnostics command. Only 0, (Return Query Data) supported.
retST	Exception status of a received command causing a NAK
RX	Received telegram. First 19 bytes in HEX format
TX	Transmitted telegram from this node. First 19 bytes in hex format.
Gpc0x	Internal troubleshooting counters. No useful information.

Note 1 This is the UART status register, bit encoded. Bit 0: Not Used. Bit 1: Overrun error. Bit 2: Parity error. Bit 3: Framing error. Bit 4: Break detected.

Note 2 Format errors have a decimal numerical value:

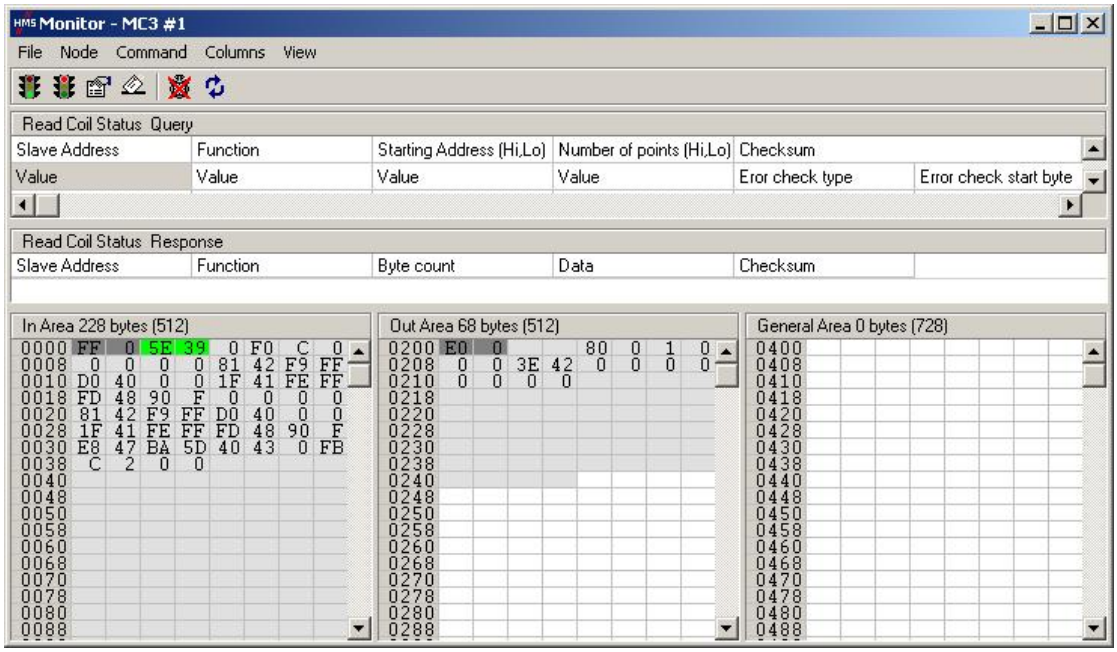
1. Unsupported Modbus command
2. Read Holding Register telegram not 6 bytes long
3. Trying to read from non-existing registers
4. Read Diagnostic telegram not 8 bytes long

5. Unsupported subfunction in Read Diagnostics telegram
6. Trying to write to non-existing registers
7. Byte count field disagrees with length field in Preset Multiple Registers command
8. Telegram length disagrees with length field in Preset Multiple Registers command
9. Trying to write to read-only registers
10. Unknown Modbus command
11. MC³ Reciever buffer overrun - more than 255 bytes in telegram.
12. Linefeed not following Carriage Return in Modbus ASCII telegram
13. Bytes received after complete telegram, before telegram interpretation (too fast).
14. Should never happen… Unknown receiver state.
15. Should never happen… Transmitter buffer overrun.

Increase Message Delimiter to see every telegram.

The telegram exchange rate typically exceeds the display update rate in the MC³ Modbus Statistics screen. If you want to see each telegram received and transmitted, you can increase the Message Delimiter time in the Sub Network parameters list the EIP configuration to 20, corresponding to 200 ms. This allows you to see all telegrams received and transmitted.

Use the ABC-PDP Node Monitor.



See ABCUM, Chapter 10. With the ABC Configurator online with the EP, click on a node and then the node monitor button. You can inspect the data, in hexadecimal format, for the selected node.

Use the integrity bit.

In the PLC, make the integrity bit (MC3[n].Ctls[0].7). the opposite of the echo bit (MC3[n].Status[0].7). If the echo bit stops toggling, communications has failed, and appropriate steps can be taken. The integrity bit can be monitored in the MC³ Data Table screen, in the EIP node monitor screen and in the CLX.